L Number	Hits	Search Text	DB	Time stamp
1	40	(high adj power) near bus	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/12/12 17:21
2	114	(low adj power) near bus	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/12/12 17:23
4	4759	snoop$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/12/12 17:19
5	2	(((high adj power) near bus) and ((low adj power) near bus)) and snoop$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/12/12 17:19
3	11	((high adj power) near bus) and ((low adj power) near bus)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/12/12 17:19
6	13431	(first or primary) near bus	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/12/12 17:22
7	13876	second$3 near bus	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/12/12 17:22
8	7084	((first or primary) near bus) same (second$3 near bus)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/12/12 17:22
9	71	(((first or primary) near bus) same (second$3 near bus)) same snoop$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/12/12 17:22
10	113358	low adj power	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/12/12 17:23

11	**6**	**((((first or primary) near bus) same (second$3 near bus)) same snoop$4) and (low adj power)**	**USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB**	**2003/12/12 17:23**

US-PAT-NO: 6085330

DOCUMENT-IDENTIFIER: US 6085330 A

TITLE: Control circuit for switching a processor between multiple **low power** states to allow cache snoops

DATE-ISSUED: July 4, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Hewitt; Larry	Austin	TX	N/A	N/A
Bunnell; James	Lafayette	CO	N/A	N/A

US-CL-CURRENT: 713/322, 711/146 , 711/3 , 713/300

ABSTRACT:

Power consumption is conserved in a computer system by, instead of forcing a processor to change from the stop clock state to a fully operational state, allowing the processor to transition from the stop clock state to the stop grant state. The stop grant state allows snoops so that the processor handles subsequent bus cycles and snoops that take place during the bus cycles.

Following the snoops, the processor transitions back from the stop grant state to the stop clock state. In one embodiment, an automatic control circuit is connected to a processor in a computer system. When the processor is in the stop clock state, the automatic control circuit responds to a bus request, not by transitioning to the fully operational state, but instead by transitioning from the stop clock state to the snoopable stop grant state in which the processor clock is operating. The automatic control circuit allows the snoop to take place then, when the snoop is complete, automatically transitions the processor back to the stop clock state.

25 Claims, 2 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 2

---------- KWIC ---------

TITLE - TI (1):
Control circuit for switching a processor between multiple **low power** states
to allow cache snoops

Brief Summary Text - BSTX (9):
C2: The CPU is in a "stop grant" state, a **low power** state in which the CPU
cache can still be snooped.

Brief Summary Text - BSTX (10):
C3: The CPU is in a "stop clock" state, a **low power** state such that the
CPU's cache cannot be snooped.

Detailed Description Text - DETX (6):
When the South Bridge 116 is instructed to place the processor 102 into the
C3 state, the South Bridge 116 responds by asserting the STPCLK# to the
processor 102. The processor 102 responds by generating a stop-grant cycle to
indicate that the processor 102 is operating in a **low power** state. The stop
grant cycle is transmitted over the Host Bus 110 and the PCI Bus 114 and
detected by the South Bridge 116. The South Bridge 116 responds by asserting
STPCPU# to the System PLL 120, causing the System PLL 120 to freeze the CPU
clock signal. With the CPU clock frozen, the processor 102 is in the **low-power**
C3 state.

Detailed Description Text - DETX (8):
In the illustrative computer system 100 utilizing C2-C3 automatic control
hardware, when the South Bridge 116 detects the request of a PCI bus cycle
(indicated by assertion of a PCI REQ# signal), the South Bridge 116 responds by
de-asserting the STPCPU# signal, waiting a suitable period of time (generally 1
millisecond or less) to stabilize an internal phase-locked loop of the
processor 102, and granting the PCI Bus 114 to the PCI master that is
requesting the bus. The processor 102 is in the C2 state, which is a

low power
state but also a state in which the PCI cycle is allowed to snoop the Internal Cache 104 via a normal protocol that is controlled in the North Bridge 108.
Once the PCI cycle is complete, the South Bridge 116 asserts the stop processor signal STPCPU# again and returns the processor 102 back in the C3 state for maximum power savings.

Detailed Description Text - DETX (17):
Once the stop processor signal STPCPU# is deasserted, a delay occurs while the phase-locked loop (not shown) that is internal to the processor 102 becomes operational. A typical duration of phase-locked loop initialization is approximately 500 .mu.s to about 1 ms. The signaling circuit waits in the stop grant state 204 until a PCI cycle complete signal takes place, indicating that the PCI bus access is complete. The PCI cycle complete signal transitions the signaling circuit from the stop grant state 204 back to the **low-power** stop clock state 206 by again asserting the stop processor signal STPCPU#. The transition from the stop grant state 204 back to the **low-power** stop clock state 206 may be immediate in some embodiments and conditions but more typically the transition to the **low-power** stop clock state 206 is delayed, for example by using a timer, to allow for interactions and arbitrations for future transactions to occur.

Detailed Description Text - DETX (20):
The illustrative computer system 100 operates differently from a conventional system to save operating power while permitting snooping. In a conventional processor operating in the C3 state, a PCI REQ# signal is a predefined resume event which transitions the processor to the C0 state. Typically, the conventional processor has a power expenditure of about ten watts in the C0 state. In the illustrative computer system 100 using stop grant (C2)--stop clock (C3) state automatic hardware control, a PCI REQ# signal received while the processor 102 is operating in the stop clock state 206 causes a transition to the stop grant state 204 in which the processor 102 has

a typical power expenditure of about one watt, advantageously reducing the power expenditure by an order of magnitude. The illustrative computer system 100 further reduces the power expenditure by remaining in the stop grant state 204 only to allow a bus cycle to take place, then returning to the very-**low-power** stop clock state 206 in which power consumption is further reduced to about 0.1 watt. In the conventional system, the processor remains in the high-power fully-operational (C0) state until an Idle determination such as a ACPI-defined command takes place.

Detailed Description Text - DETX (21):

Accordingly, the C2-C3 automatic control hardware advantageously controls the processor 102 to predominantly operate in the very-**low-power** (0.1 watt) stop clock state 206 and to enter the moderate-power (1 watt) stop grant state 204 only momentarily while servicing a PCI request before returning to the stop clock state 206. In contrast, a processor operating under ACPI standards converts for an indefinite time to the high-power (10 watt) C0 state upon an occurrence of a PCI request, and remains in the high-power C0 state until an ACPI-defined command is issued.

Claims Text - CLTX (2):

a processor including an internal cache, the processor operating in a plurality of power states including a fully operational state in which the processor is fully operational, a stop grant **low-power** state in which the internal cache is snooped, and a stop clock **low-power** state in which the internal cache is not snooped;

Claims Text - CLTX (4):

a state control circuit coupled to the processor and coupled to the clock generator, the state control circuit controlling the processor while operating in the stop clock **low-power** state to transition to the stop grant **low-power** state in response to a bus request signal, snoop the processor internal cache, and return from the stop grant **low-power** state to the stop clock

low-power
state following the snoops; and wherein

Claims Text - CLTX (6):
the processor transitions from the filly operational state to the stop clock
low-power state on receipt of an idle determination; and

Claims Text - CLTX (7):
the state control circuit transitions the processor from the stop grant
low-power state to the stop clock **low-power** state on an occurrence of a bus
cycle complete signal.

Claims Text - CLTX (12):
the state controller asserts the STPCLK# signal and asserts the STPCPU#
signal in the stop clock **low-power** state; and

Claims Text - CLTX (13):
the state controller deasserts the STPCPU# signal on entry into the stop
grant **low-power** state.

Claims Text - CLTX (18):
the stop grant **low-power** state is a C2 state;

Claims Text - CLTX (19):
the stop clock **low-power** state is a C3 state; and

Claims Text - CLTX (23):
a processor coupled to the first bus and including an internal cache, the
processor operating in a plurality of power states including a fully
operational state in which the processor is fully operational, a stop grant
low-power state in which the internal cache is snooped, and a stop clock
low-power state in which the internal cache is not snooped;

Claims Text - CLTX (27):
an interface coupled to the processor via the **second bus and the first bus,**
the interface including a state control circuit coupled to the processor and
coupled to the clock generator, the state control circuit controlling the

processor while operating in the stop clock **low-power** state to
transition to
the stop grant **low-power** state in response to a bus request signal,
snoop the
processor internal cache, and return from the stop grant **low-power**
state to the
stop clock **low-power** state following the **snoops**; and wherein

Claims Text - CLTX (29):
the processor transitions from the fully operational state to the
stop clock
low-power state on receipt of an idle determination; and

Claims Text - CLTX (30):
the state control circuit transitions the processor from the stop
grant
low-power state to the stop clock **low-power** state on an occurrence of a
bus
cycle complete signal.

Claims Text - CLTX (35):
the state controller asserts the STPCLK# signal and asserts the
STPCPU#
signal in the stop clock **low-power** state; and

Claims Text - CLTX (36):
the state controller deasserts the STPCPU# signal on entry into the
stop
grant **low-power** state.

Claims Text - CLTX (41):
the stop grant **low-power** state is a C2 state;

Claims Text - CLTX (42):
the stop clock **low-power** state is a C3 state; and

Claims Text - CLTX (44):
11. A method of operating a computer system including a processor
with an
internal cache, the processor operating in a plurality of power states
including a fully operational state in which the processor is fully
operational, a stop grant **low-power** state in which the internal cache
is
snooped, and a stop clock **low-power** state in which the internal cache
is not
snoopable, the method comprising:

Claims Text - CLTX (46):
transitioning the processor from the fully operational state to the stop
clock **low-power** state on an occurrence of an idle determination, generation of
an external processor clock being stopped in the stop clock **low-power** state;

Claims Text - CLTX (47):
transitioning the processor from the stop clock **low-power** state to the stop
grant **low-power** state on an occurrence of a memory access request, generation
of the external processor clock being enabled in the stop grant **low-power** state
and an internal processor clock being stopped in the stop grant **low-power**
state;

Claims Text - CLTX (48):
while the processor is operating in the stop grant **low-power** state, snooping
the processor internal cache; and

Claims Text - CLTX (49):
transitioning from the stop rant **low-power** state to the stop clock **low-power**
state following the snoops.

Claims Text - CLTX (51):
transitioning the processor from the stop grant **low-power** state or the stop
clock **low-power** state to the fully operational state on an occurrence of a
predefined resume event.

Claims Text - CLTX (53):
transitioning the processor from the stop grant **low-power** state or the stop
clock **low-power** state to the fully operational state on an occurrence of an
interrupt.

Claims Text - CLTX (55):
transitioning the processor from the stop grant **low-power** state to the stop
clock **low-power** state on an occurrence of a bus cycle complete signal.

Claims Text - CLTX (57):
on entering the stop grant **low-power** state, deasserting a STPCPU# signal
stopping generation of the external processor clock to the processor; and

Claims Text - CLTX (58):
also on entering the stop grant **low-power** state, enabling a bus cycle.

Claims Text - CLTX (61):
the stop grant **low-power** state is a C2 state;

Claims Text - CLTX (62):
the stop clock **low-power** state is a C3 state; and

Claims Text - CLTX (64):
17. A method of operating a computer system including a processor with an
internal cache, the processor operating in a plurality of power states
including a fully operational state in which the processor is fully
operational, a stop grant **low-power** state in which the internal cache is
snooped, and a stop clock **low-power** state in which the internal cache is not
snooped, the method comprising:

Claims Text - CLTX (66):
transitioning the processor from the fully operational state to the stop
clock **low-power** state on an occurrence of an idle determination;

Claims Text - CLTX (67):
transitioning the processor from the stop clock **low-power** state to the stop
grant **low-power** state on an occurrence of a bus request signal;

Claims Text - CLTX (68):
on entering the stop clock **low-power** state, asserting a STPCLK# signal
causing the processor to enter a Stop Grant **low-power** state during which an
internal clock of the processor is stopped; and

Claims Text - CLTX (69):
also on entering the stop clock **low-power** state, asserting a STPCPU# signal

stopping generation of an external processor clock to the processor.

Claims Text - CLTX (70):
18. A method of operating a computer system including a processor with an
internal cache, the processor operating in a plurality of power states including a fully operational state in which the processor is fully operational, a stop grant **low-power** state in which the internal cache is
snooped, and a stop clock **low-power** state in which the internal cache is not
snooped, the method comprising:

Claims Text - CLTX (72):
transitioning the processor from the fully operational state to the stop
clock **low-power** state on an occurrence of a predefined command, the stop clock
low-power state being a state in which an external clock supplied to the
processor is stopped;

Claims Text - CLTX (73):
transitioning the processor from the stop clock **low-power** state to the
snoopable stop grant **low-power** state on an occurrence of a memory access
request, the stop grant **low-power** state being a state in which the external
clock supplied to the processor is running and an internal clock in the processor is stopped; snooping the internal cache in the processor in the stop
grant **low-power** state; and

Claims Text - CLTX (74):
after the snoop is complete, transitioning the processor from the stop grant
low-power state to the stop clock **low-power** state.

Claims Text - CLTX (76):
transitioning the processor from the stop-grant **low power** state to the stop
clock **low-power** state occurs on an occurrence of a bus cycle complete signal.

Claims Text - CLTX (78):
transitioning the processor from the stop grant **low-power** state or the stop
clock **low-power** state to the fully operational state on an occurrence

of a
predefined resume event.

Claims Text - CLTX (80):
transitioning the processor from the stop grant **low-power** state or the stop
clock **low-power** state to the fully operational state on an occurrence of an
interrupt.

Claims Text - CLTX (82):
on entering the stop grant **low-power** state, deasserting a STPCPU# signal to
resume generation of the external clock to the processor; and

Claims Text - CLTX (83):
also on entering the stop grant **low-power** state, enabling a bus cycle.

Claims Text - CLTX (86):
the stop grant **low-power** state is a C2 state;

Claims Text - CLTX (87):
the stop clock **low-power** state is a C3 state; and

Claims Text - CLTX (90):
25. A method of operating a computer system including a processor with an
internal cache, the processor operating in a plurality of power states
including a fully operational state in which the processor is fully
operational, a stop grant **low-power** state in which the internal cache is
snooped, and a stop clock **low-power** state in which the internal cache is not
snooped, the method comprising:

Claims Text - CLTX (92):
transitioning the processor from the fully operational state to the stop
clock **low-power** state on an occurrence of a predefined command;

Claims Text - CLTX (93):
transitioning the processor from the stop clock **low-power** state to the
snoopable stop grant **low-power** state on an occurrence of a bus request signal;

Claims Text - CLTX (94):
allowing a snoop to take place in the stop grant **low-power** state;

Claims Text - CLTX (95):
when the snoop is complete, transitioning the processor from the stop grant
low-power state to the stop clock **low-power** state;

Claims Text - CLTX (96):
on entering the stop clock **low-power** state, asserting a STPCLK# signal
causing the processor to enter a Stop Grant **low-power** state during which an
internal clock of the processor is stopped; and

Claims Text - CLTX (97):
also on entering the stop clock **low-power** state, asserting a STPCPU# signal
stopping generation of an external processor clock to the processor.

US-PAT-NO: 5796977

DOCUMENT-IDENTIFIER: US 5796977 A

TITLE: Highly pipelined bus architecture

DATE-ISSUED: August 18, 1998

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Sarangdhar; Nitin V.	Portland	OR	N/A	N/A
Singh; Gurbir	Portland	OR	N/A	N/A
Lai; Konrad	Aloha	OR	N/A	N/A
Pawlowski; Stephen S.	Beaverton	OR	N/A	N/A
MacWilliams; Peter D.	Aloha	OR	N/A	N/A
Rhodehamel; Michael W.	Beaverton	OR	N/A	N/A

US-CL-CURRENT: 709/1, 711/141 , 711/146 , 712/220

ABSTRACT:

A computer system incorporating a pipelined bus that maintains data coherency, supports long latency transactions and provides processor order is described. The computer system includes bus agents having in-order-queues that track multiple outstanding transactions across a system bus and that perform snoops in response to transaction requests providing snoop results and modified data within one transaction. Additionally, the system supports long latency transactions by providing deferred identifiers during transaction requests that are used to restart deferred transactions.

16 Claims, 15 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 15

---------- KWIC ---------

Detailed Description Text - DETX (5):

The signal lines and logic of the system bus 20 is implemented using Gunning Transceiver Logic (GTL) from Xerox.RTM. Corporation which provides **low power** consumption and electromagnetic interference (EMI). The use of this technology allows up to eight agents to be coupled to system bus 20 while still maintaining a bus clock speed of up to 100 MHz. Various embodiments incorporate various clock speeds including 33.3 MHz, 44.4 MHz, and 66.7 MHz although other clock speeds may also be used. These clock speeds allow the invention to be incorporated into computer systems having various hardware capabilities.

Detailed Description Text - DETX (43):

FIG. 11 is a timing diagram illustrating the situation where two sequential (i.e., back-to-back) pipelined transactions request the same data location, and therefore how data coherency is preserved while the pipeline is maintained. A **first bus** agent ("A1") initiates a first invalidate line transaction at clock cycle T2 by asserting a logic-low ADS#. An invalidate line transaction signals all other memory agents on the system bus to place this data location into an Invalid state because A1 wishes to modify this data. Three cycles later, a **second bus** agent ("A2") makes a request for the same address also indicated by a logic low on ADS#. In the preferred embodiment of the system bus incorporating the invention, A1 observes this second request and determines that it is for the same data location it requested in the first transaction. Since A1 assumes ownership of this data location after the **Snoop** Phase of the first transaction, A1 will provide the proper **snoop** result, asserting a HITM# during the **Snoop** Phase of the second transaction.

US-PAT-NO: 6581124

DOCUMENT-IDENTIFIER: US 6581124 B1

TITLE: High performance internal bus for promoting design reuse in north bridge chips

DATE-ISSUED: June 17, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Anand; Vishal	Fremont	CA	N/A	N/A

US-CL-CURRENT: 710/305, 710/100 , 710/107 , 710/113 , 710/241

ABSTRACT:

In an example embodiment, an apparatus providing communication in a computer system, comprises, a plurality of modules each having a master port and a slave port A secondary bus is shared between the plurality of modules for transmitting data and address information between a master port and a slave port of two modules. A bridge circuit coupled to the plurality of modules and the secondary bus, individually grants modules of the plurality of modules, access to the secondary bus. The bridge circuit establishes point-to-point communication paths between a master port and a slave port of two modules of the plurality of modules, for communicating handshake signals between them, and controls address and data phases between modules; two address phases can be outstanding simultaneously. The bridge circuit forwards address and data phases from one module to another module of the plurality of modules; the plurality of modules only interface with the bridge circuit.

25 Claims, 43 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 43

---------- KWIC ---------

Detailed Description Text - DETX (22):
The following signal descriptions are one embodiment, in accordance with the present invention, of the definitions of the central services signals. Signal Name: Clock (clk) 360 Output: 1 State Meaning: Clock input for the module. All timing on the secondary bus is referenced to this clock. Timing: Free running in normal mode. Can be held in logic level low if both `qreq_` and `qack_` are asserted, which are both described below. Signal Name: Reset (reset_) 362 Output: 1 State Meaning: Assertion of this signal indicates that modules should enter idle state and all inputs should be ignored. Timing: May be asserted or de-asserted on any cycle synchronous to `clk`. Signal Name: Quiescent Clock (qclk) 364 Output: 1 State Meaning: Used as a clock to reference the signals `qreq_` and `qack_`, which are described below. Timing: Not Applicable. Signal Name: Quiescent Request (qreq_) 366 Output: 1 State Meaning: Assertion of this signal indicates that the module should terminate or pause all activity so that the chip may enter a quiescent (or a **low power**) state. Timing: May be asserted or de-asserted on any cycle synchronous to `qclk` 364. Signal Name: Quiescent Acknowledge (qack_) 368 Output: 0 State Meaning: This signal indicates that the module has ceased all activity and is ready to enter into a quiescent state. Timing: May be asserted or de-asserted on any cycle synchronous to `qclk` 364.

Detailed Description Text - DETX (50):
Within the present embodiment, power management on secondary bus 216 of FIG. 2 is achieved by using signals `qclk` 368, `qreq_` 364, and `qack_` 366. These signals are used by a power management unit to bring the chip into a **low power** state. On sampling a `qreq_` 364, a module should complete all outstanding transactions, flush its buffers and stop all external arbitration. On completion of all these events, the module should assert `qack_` 366. On sampling `qack_` 366 from a module, the power management unit can shut off all the clocks going to the module. It should be appreciated that the power management unit is responsible for implementing a clock gating scheme.

Wake up from the power down state can be triggered by either the module or the power management controller. The various possible stages are shown in FIGS. 14-16.

Detailed Description Text - DETX (77):

Referring to FIG. 28, the modules connected to one embodiment of a **primary bus** in accordance with the present invention, are a CPU slave module 2102, a CPU master module 2104, a memory module 2114, and a bridge module 214. FIG. 28 is a block diagram showing the communication traffic which is possible between the various modules over the **primary bus**. It should be appreciated that arrows 2802-2808 represent data transfers between the modules, while arrows 2810-2818 represent address transfers between the modules. Bridge module 214 forwards the cycles from the modules on **secondary bus** 216, of FIG. 21, to memory module 2114 directly (if no **snooping** is required) or through CPU master interface 2104 (after the **snoop** is complete). Memory module 2114 can get addresses from three modules and data from two modules. Instead of sending the address directly to memory module 2114, it can be routed through bridge module 214. This way memory module 2114 has a single port for address and two ports for data. FIG. 29 is a block diagram showing another interconnect scheme of the **primary bus**. The bus protocol used to communicate between the modules of the **primary bus** can be the same as the **secondary bus** to start with, and later can be adapted as the bandwidth requirement changes.